CLAIMS

1. A method of generating a development project including at least a matrix switch and one or more adjacent objects, the method comprising:

establishing an initial rendering of the development project; and

negotiating buffer size and attribute characteristics between an input/output of the matrix switch and an input/output of adjacent objects, wherein the negotiated buffers are utilized to communicate media content between the matrix switch and the adjacent buffers by sharing a common buffer between the inputs and the outputs.

2. A method according to claim 1, further comprising

modifying input/output associations between objects in the initial rendering of the development project based at least in part on the negotiation.

3. A method according to claim 2, wherein the input/output associations are communicative connections through one or more buffers.

4. A method according to claim 1, wherein the initial rendering of the development project included a separate buffer for each input and output of each object within the project, some of which are replaced with a single buffer shared between select input(s) and output(s) based, at least in part, on the negotiation.

5. A method according to claim 1, wherein the matrix switch attempts to be an allocator for buffers shared with each of its input(s) and output(s).

6. A method according to claim 5, wherein if the matrix switch cannot be an allocator for one or more of its input(s) or output(s), such input(s) and output(s) do not share a common buffer with objects coupled thereto.

7. A method according to claim 6, wherein memory copy operations are utilized to communication information to/from input(s) and/or output(s) of the matrix switch for which the switch is not the allocator.

8. A method according to claim 6, wherein the development project is a media processing project rendered as a filter graph of processing chains.

9. A storage medium comprising a plurality of executable instructions which, when executed, implement a method of claim 1.

10. A computing system comprising:

a storage medium having stored therein a plurality of executable instructions; and

an execution unit, coupled to the storage medium, to execute at least a subset of the plurality of executable instructions to implement a method according to claim 1.

11. A development system comprising:

one or more processing chains; and

a matrix switch, coupled to the one or more processing chains, to recursively pass content received from the one or more processing chains through one or more processing objects to implement a development project, wherein the matrix switch negotiates buffer size and attributes between the matrix switch and adjacent objects, wherein the negotiated buffers are utilized to communicate media content between the matrix switch and the adjacent buffers without requiring a buffer copy operation.

12. A development system according to claim 11, wherein each of the objects comprising the one or more processing chains attempt to negotiate buffer size and attribute characteristics in order to facilitate a shared buffer for communicating information between the objects of the processing chain.

13. A development system according to claim 12, wherein the objects establish shared buffers between an input of one object and the output of an upstream object upon negotiating mutually acceptable buffer size and attribute characteristics.

14. A development system according to claim 11, wherein the development project is established by a render engine, exposed from an operating system executing on a computing system implementing the development system.

15. A development system according to claim 14, wherein the render engine facilitates negotiation between objects of the processing chains of buffer size and attribute requirements, and establishes a shared buffer for communicating content between objects when an agreement as to the requirements is achieved.

16. A development system according to claim 11, wherein the matrix switch negotiates to be an allocator of buffers between the matrix switch and any object coupled to its input and output to facilitate communication between the matrix switch and external objects as well as between its input(s) and output(s) without the need for a memory copy operation.

17. A development system according to claim 16, wherein if the matrix switch is not able to be an allocator of a buffer for an input or an output of the matrix switch, a memory copy operation will be required to communicate with that input or output.

18. A development system according to claim 17, wherein a memory copy operation is required to communicate information to/from an matrix switch input and/or output for which the matrix switch is not an allocator of a buffer associated with that input and/or output, even if the communication is internal to the matrix switch itself.

19. A matrix switch object comprising:

a dynamically determined number of inputs to receive content from one or more processing chains; and

a dynamically determined number of outputs, selectively coupling one or more of the dynamically determined inputs to one or more of the dynamically determined outputs, wherein the matrix switch negotiates with objects coupled to each of the dynamically determined inputs and outputs for buffer size and attribute requirements to facilitate communication between objects and within the matrix switch using a shared buffer of agreed upon size and attribute characteristics.

20. A matrix switch object according to claim 19, wherein if the matrix switch cannot negotiate an agreed upon buffer size and attribute characteristics between an input/output and an object coupled to the input/output, communication with the input/output is performed using a memory copy operation.

21. A matrix switch object according to claim 20, wherein an input/output coupling the object to the input/output of the matrix switch each have an independent buffer, wherein communication occurs between the object and the matrix switch by copying content from one buffer to another buffer.

22. A matrix switch object according to claim 19, wherein communication between the input/output of the matrix switch and any other input/output, internal or external to the matrix switch is performed using a memory copy operation.

23. A matrix switch according to claim 20, wherein if an input/output of the matrix switch and an input/output of an object coupled to the input/output of the matrix switch do agree upon buffer size and attribute requirements, communication between the object and the matrix switch will be through a shared buffer coupling the input/output of the object to the input/output of the switch.

24. A matrix switch according to claim 23, wherein communication between the input/output of the matrix switch and a second input/output of the matrix switch will be through a shared buffer, unless the second input/output does not adhere to the agreed upon buffer size and attribute requirements.

25. A matrix switch according to claim 20, wherein matrix switch identifies buffer size and attribute requirements of all objects coupled to an input/output of the matrix switch, and attempts to negotiate a common buffer size and attribute requirement for all switch input(s) and output(s).

26. A matrix switch according to claim 19, further comprising a plurality of buffers, shared between the dynamically determined inputs and the dynamically determined outputs to buffer processed media content for subsequent use by objects coupled to the matrix switch.